final draft

Lecture 9 - Scripting and domain-specific languages

"There is no programming language that will prevent programmers from making bad programs." - Larry Flon

- how can we write bespoke code per game object without adding complexity to engine/game code?
- how can we improve the compile time of game code?
- how can we make it easy for the community to contribute their additions to the game without changing any source code?

Last Week Recap

- AI

Interpreting vs Compiling

- Interpreting runs slower but no compilation times
- Compiled runs faster but need to compile before running
- At type level there can be major differences (dynamic vs static typing)
- Scripts can support both

Importance of Scripting

- Hardcoding each game object behaviour is messy and cannot be effectively maintained
- Not using scripting and changing source code affects compilation times
- With scripts changes can be tested instantly at runtime

Scripting Languages and Advantages

Python, JS, Lua, ...

- Allow updating game behaviour without changes to the engine / game source code.
- Allow calling engine functions from scripts.
- Allow user content.
- Reusable.

Usage

Many games and engines use scripting:

- Godot, Unreal, Unity
- The Elder Scrolls series, Fallout series, GTA series, ...

Scripting

Typically a subset of a full programming language. In a script:

```
spawn("door", x, y)

...

fire_event(door_opened_event)

...

engine.current_time
```

Activity

MMORPG script example. Go to Inn script

1. What are the comment indicators?
2. What does function `mes` do?
3. Overall, what does the script achieve?

Scripting Concepts

- Lexer / tokenizer
- Parser

Lexer / Tokenizer

1. Goes through each literal string and converts to a type in the language.
2. Fails if there are syntax issues

Parser

1. Takes tokens from the tokenizer and evaluates expressions from top to bottom.

Demo Script Implementation

Example: suppose we want to add a new gameplay element to an existing weapon (assault rifle) in a game.

In pseudo code:

```
type DamageModifierOnReload extends EventScript {

    onEvent(event) {
        if (event.type == "WEAPON_RELOAD") {
            event.owner.applyEffect(30, "DAMAGE_MODIFIER",
        }
    }
}
```

Activity

1. Open and read Hello World page for Skyrim Papyrus scripting engine.
2. Consider a game of your choice (examples: Minecraft, TESV Skyrim, Fallout, etc.)
3. In pseudo code (on paper), construct a script that implements a new gameplay element.

Conclusion

- Scripting is a very powerful addition to any engine
- If custom, can be difficult to develop / maintain – don't create yet another programming language

Tutorial

Assignment work